Framework for Heterogeneous Multi-Domain Swarms to Improve Maritime Situational Awareness

Stefan Kampmann*, Sarah Barnes**, Maurice Stephan
Institute for the Protection of Maritime Infrastructures
German Aerospace Center (DLR)
Bremerhaven, Germany
stefan.kampmann@dlr.de*, sarah.barnes@dlr.de**, maurice.stephan@dlr.de

***Abstract*—This paper describes current development efforts for a Command and Control (C2) framework for utilizing heterogeneous multi-domain robotic swarms. The NEtworked X-domain Uncrewed Systems (NEXUS) framework aims at supporting the operations of dynamically composed heterogeneous robot swarms primarily consisting of Uncrewed Aerial Vehicles (UAVs), Uncrewed Surface Vehicles (USVs), and Uncrewed Underwater Vehicles (UUVs). NEXUS aims at enabling and accelerating research and deployment efforts and utilization of Sensor Carrier Systems (SCSs) for an improved maritime situational awareness. Especially in the context of surveying and protecting critical maritime infrastructure, but also in the context of scientific exploration NEXUS could serve human operators as a highly flexible tool for researching robotics topics, gathering situational information and aggregating data into a cohesive data product.**

***Index Terms*—command and control, ground station, multi-domain robotics, human-swarm interface, autonomy, framework, middleware, systems engineering.**

I. Introduction

The recent increase in tensions and incidents around pipelines and data cables in the Baltic Sea [1], as well as the increased importance of uncrewed vehicles (UVs) as actors in nations' defence and security concepts [2], [3] shows a need for enhanced security of critical maritime infrastructures capable of using UVs and reacting to dynamic threats.

UAV-centric swarms are increasingly used for situational awareness in complex environments [4], providing spatial overviews and close-up views from a birds-eye perspective. Multi-domain heterogeneous swarms consisting of UAVs, USVs, and UUVs may impose a similar benefit for maritime situational awareness by combining sensor data into one full-view data product, additionally covering water surface and underwater viewpoints. Use cases could range from observation operations as part of critical infrastructure control, over maintenance and exploration missions with data aggregation and fusion, to supporting aid measures by providing information [5]. However, there exists no common open-source framework for coordinating such heterogeneous swarms with the general focus of automating mission generation and management.

As a concept, we propose a C2 framework able to handle individual and swarm-based robot control and coordination, consisting of a domain agnostic ground station (GS) and drop-in mobile SCSs, where individual SCSs can be dynamically configured to work together as a swarm. This framework concept allows access to the capabilities and sensor data of individual SCSs as well as swarms in arbitrary compositions. It also aims at abstracting the individual vehicles' technical details to allow rapid mission definition and automated mission and vehicle management. A single operator can oversee multiple missions and vehicles at once, while multiple operators can use the system in parallel. Furthermore, this framework will expose a data interface to provide external data processing pipelines with sensor data gathered from any of the SCSs deployed at that moment.

Furthermore, this paper will introduce the UI for this framework as the first step towards the framework's implementation.

II. State of the Art

End-to-end C2 capability for any mobile robot or swarm requires a complete toolchain, including the (graphical) user interface (GUI), data processing, and communications, down to individual vehicle control. The following frameworks and tools solve at least one of these aspects. Since the resulting framework will support a wide variety of research and security-related applications, all data needs to be managed transparently and vendor-lock-in is crucial to avoid. As such, only open-source projects are considered.

A. Toolchains

The Laboratório de Sistemas e Tecnologia Subaquática **(LSTS)-Toolchain** [6], [7] is comprised of four tools providing full command, control, communications, and intelligence/information (C3I) capabilities:

- **Ripples:** A web interface for centralized asset supervision and control, improving situational awareness for operators but also external personnel [8], [9].
- **Neptus:** The distributed C2 infrastructure, utilized for mission planning, simulation, execution, and post-mission analysis [10], [11].
- **IMC:** The communication protocol defining common messages for all modules to communicate with [12], [13].
- **DUNE:** Embedded software running on the robot units, taking over control, reading sensors, and communicating with Neptus [14].

While this toolchain provides a similar core functionality, it lacks capabilities in automated and self-managed mission generation and execution for individual or heterogeneous multi-domain swarms, as well as automated mission and SCS management. Such capabilities are necessary to cover maritime infrastructures in vast and/or complex maritime environments. Especially when utilizing many different SCSs in task-specific heterogeneous swarms at the same time with a small operator team and minimal human risk. A framework capable of supporting such operations further enables research areas which the LSTS-toolchain is not optimized for, for example environment-optimized mission plan generation or self-organizing fully autonomous SCS. Furthermore, the operators within NEPTUS are not distinguished: As soon as a NEPTUS instance, or a possible attacker mimicking one, is on the same network as other devices, any SCS reachable through either IRIDIUM, GMS, WIFI, or an acoustic modem can be commanded. Also, any vehicle capable of logging into the network can be commanded as well. An exact pre-definition and identification of allowed operators and vehicles is missing in the LSTS toolchain. Lastly, individual missions are treated as workspaces in NEPTUS, consisting of individual plans, consisting of individual maneuvers. Maneuvers within plans cannot be reordered once added. This may lead to clutter in a single workspace and loading times due to loading entire workspaces into NEPTUS, when an operator does not properly manage their workspaces.

The Mission Oriented Operating Suite (MOOS) with Interval Programming (IvP) **MOOS-IvP** is a collection of software modules providing autonomy for marine vehicles [15]. The project is situated at MIT as part of the Laboratory for Autonomous Marine Sensing Systems (LAMSS) [16] and partially maintained at the Oxford Robotics Institute (ORI) [17]. It is a complex toolchain, covering core autonomy functionalities, simulation, vehicle behavior, and mission control amongst others. With its highly modular design, it can be used in many different marine scenarios. However, it lacks C2 capability for UAVs and a GUI allowing quick mission definitions. Parallel utilization with multiple operators at the same time is also not implemented.

B. Ground Control Station (GCS)

GCSs comprise all components on the ground, facilitating data processing, mission planning, and vehicle control interfaces. Many open-source systems utilize the ArduPilot [18] or PX4 [19] autopilots. Both provide autopilot software for UAVs, but also for other vehicle types, such as UUVs.

ArduPilot provides the GCS **Mission Planner** [20] for mission planning, observing, and remote vehicle control. While PX4 does not provide a specific GCS, **Q-Ground Control** [21] can be used with both PX4 and ArduPilot for C2 capabilities. Both GCSs utilize the MAVLink communications protocol [22]. However, multi-vehicle functionality is heavily limited, and there is no automatic mission generation, swarming, nor further data processing or automated mission and SCS management.

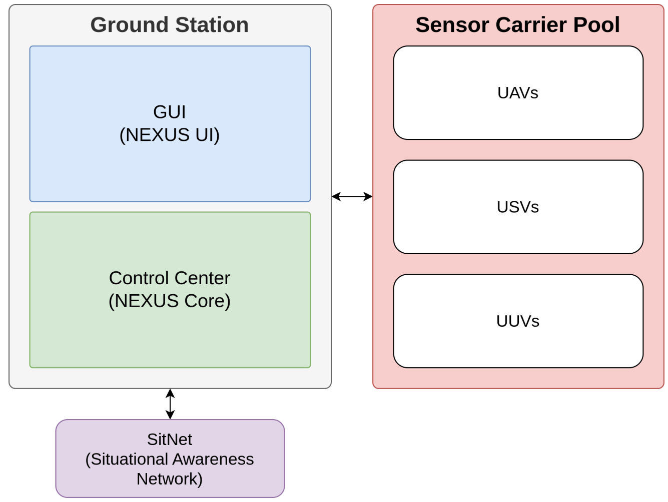


Fig. 1: On the highest level, NEXUS is made up from two sections: The Ground Station components *NEXUS UI* and *NEXUS Core* for defining, controlling, and monitoring missions and vehicles. And, the Sensor Carrier Pool made up from all SCSs known to NEXUS Core. Additionally, data sources and sinks can be connected to NEXUS Core through a data interface, here exemplary named as Situational Awareness Network SitNet. Communication between the NEXUS core and individual SCS of the SCP utilizes the IMC [13] messaging protocol.

C. Robot Control Frameworks

There are several options for controlling individual robots with embedded software. The most popular one is the **Robot Operating System 2 (ROS2)** [23]. It provides a large library of functionalities to program all kinds of robots from the ground up. Alternatives like the **Mobile Robot Programming Toolkit (MRPT)** [24] or **Yet Another Robot Platform (YARP)** [25] typically provide interfaces to ROS2 and are not further considered due to the scope of ROS2. It is possible to program a low level controller for each type of SCS, directly interfacing with the sensors and actors. However, the autopilot hardware and software typically exists on off-the-shelf systems. At most, an interface between an existing autopilot, an SCS's payload and a C2 infrastructure needs to be defined. For this, ROS2 provides ample functionality.

III. Framework Architecture

NEXUS focuses on explicit system and operator definitions and distinguishes between operators, while remaining easy to access. On the highest level, NEXUS's framework is comprised of two components: The GS and the Sensor Carrier Pool (SCP), see Fig. 1. The GS takes on all SCS, mission, and data management tasks and interfaces with human operators. It consists of the Human-in-the-loop (HITL) GUI NEXUS UI, and the control center NEXUS core. The SCP consists of individual SCSs operating in the domains air, water-surface, or underwater. SCSs specific to the domain land are not considered at this time. Additionally, the framework exposes

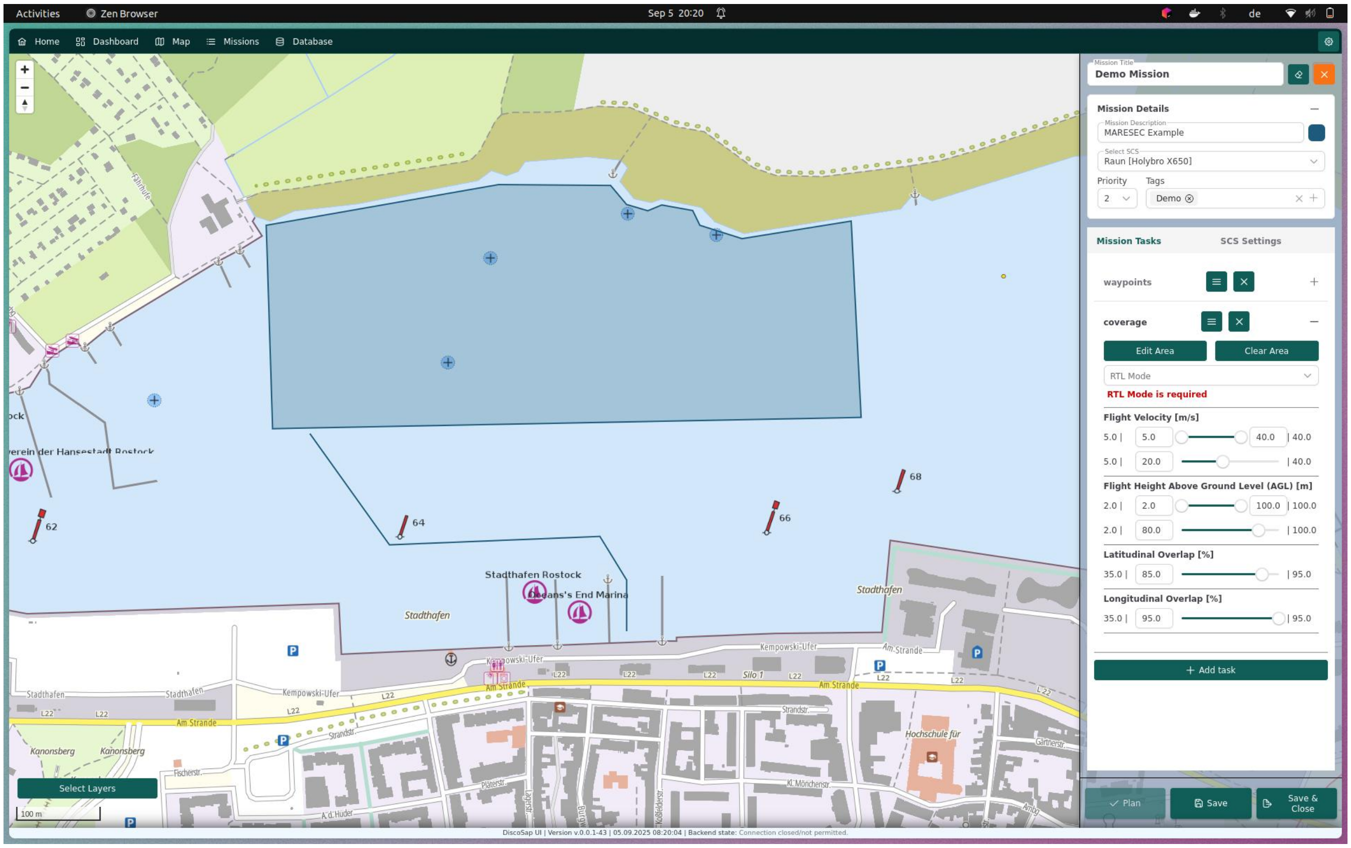


Fig. 2: The NEXUS UI used as the C2 interface to the NEXUS framework. Example: Currently editing mission 'Demo Mission' (blue) with opened mission edior. The mission currently consists of two tasks, where task-specific parameters can be specified. Each mission may be comprised of multiple tasks, such as an area survey or waypoint task. Mission can be added to this map/editor view from the missions list. The layer manager (collapsed, button in the bottom-left) allows focusing, hiding/showing, or removing individual mission form the map.

an interface to external data sources and sinks, exemplary depicted in fig. 1 as SitNet. A database is used to explicitly define vehicles and their operational capabilities, while useres are specified through user authentication as the NEXUS core is deployed on a typical server infrastructure.

A. NEXUS UI

To facilitate multiple SCSs operating in parallel in different domains with multiple operators defining and monitoring missions at the same time, NEXUS UI provides the HITL interface, see fig. 2. It is a browser-based web application developed with Angular [26] which enables operation-system-independent access to the framework's capabilities. Multiple views are available to satisfy an operator's needs:

- **Dashboard:** Mission-centric view of sensor data and SCS state and meta data. For each mission, the dashboard can be individually configured to show wanted sensor data.
- **Map:** Simplified mission definition with a mission editor and situational awareness by showing processed and geo-located data.
- **Mission list:** List of all available missions independent of state (e.g. planned, running, or completed) and mission-previews with the option to edit mission details.
- **Remote control:** A view specialized on remote controlling a single SCS. The specific sensor data to be shown here can be configured as well. SCSs can be taken over even in missions. This provides explicit human control and intervention capabilities, especially to mitigate critical situations an SCS may not be able to handle itself.
- **Database:** An overview of the SCS and missions saved in the database. This overview allows interaction with the missions in the forms of editing, duplicating, or removing missions. To ensure the SCS are used as intended their database overview is defined as read-only.

NEXUS UI provides live data based on the specific view's configuration, enabling explicit control over all of the framework's SCSs and missions. Furthermore, sources like web-map-services or slippy map tiles servers can be dynamically added to the map, allowing the operator to modify the map's appearance. For example, data overlays providing additional information like the current weather, or geozones

DOI: 10.5281/zenodo.17120080

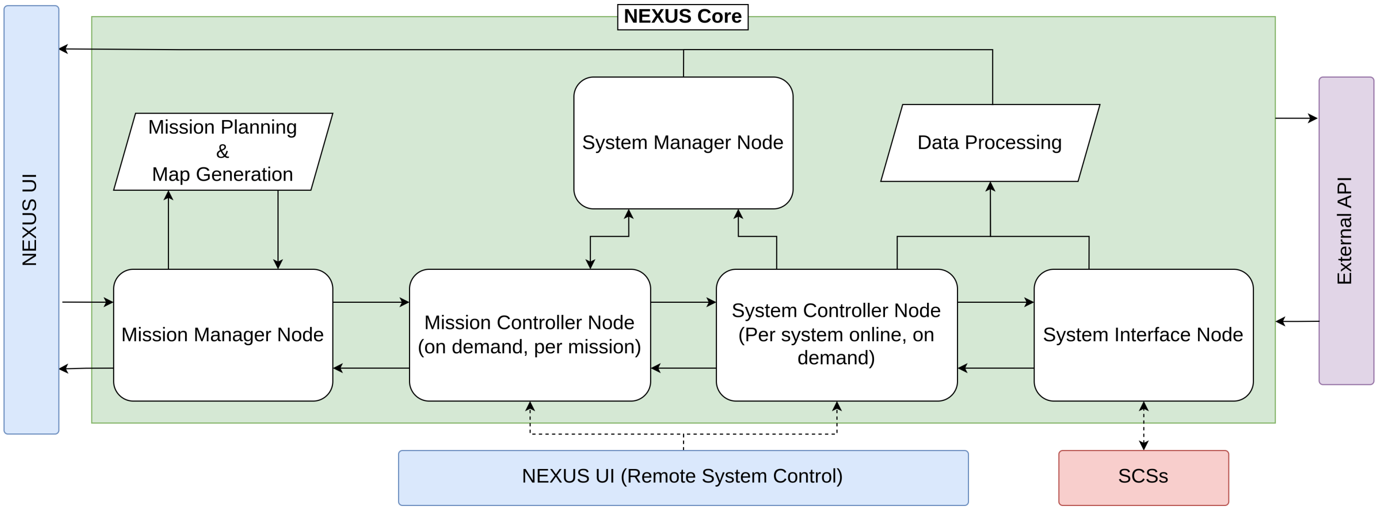


Fig. 3: The NEXUS Core in a high-level overview. When an SCS (or system) is remote controlled, the NEXUS UI has direct influence on the mission manager and the specific SCS controller node of the controlled system.

for flying UAVs increase the operator's awareness for the environmental conditions while planning missions.

B. NEXUS Core

NEXUS Core is a collection of processing nodes implemented in C++ which communicate through a custom middleware abstracting FastDDS [27]. Abstracting the messaging framework allows changing it to a different one, such as ZeroMQ [28], without large changes to the NEXUS Core's code base if the need arises at a later time. This prevents vendor lock in. The core's main functionalities are:

- **Mission management:** Keeping track of current missions and their states
- **Mission planning:** Determines initial mission paths and actions required to service mission requests.
- **Map generation:** Utilizes SCS data and external data sources to maintain a master map. SCS-specific map versions are provided to each SCS on deployment to allow SCS-based local mission and behavior planning.
- **Mission controller:** Manages one individual mission per existing mission contorller to enable interventions if necessary and clearly separate SCS from missions.
- **SCS controller:** Manages an individual SCS's deployment procedures, monitors system state, and allows direct remote control if needed.
- **SCS management:** Keeps track of all SCSs registered in the system and their mission assignment.
- **SCS interface:** Node to distribute the data flow to and from the SCSs.
- **Data processing:** Necessary data processing to provide nodes with sensor and SCS state data where required. Also processes the data from and to the external API.

Fig. 3 shows an overview of the nodes and processing pipelines involved in the NEXUS Core. When a mission is requested the mission management node utilizes the mission planning and map generation pipelines to create a mission plan. The mission planning pipeline takes into account the individual SCSs' capabilities, selects suitable SCSs for a mission, determines the waypoints and waypoint parameters to service the mission request, and hands the mission plan back to the mission manager.

The mission manager creates a mission controller node to work through this mission plan. This mission controller binds the selected SCSs' controllers to itself when they become available, blocking their SCSs from being allocated to a different mission. This prevents control conflicts. The SCS manager provides the required state information for this process to the mission controller. When all SCSs are bound to the mission controller they are deployed to execute the mission plan.

While the mission-specific robot swarm is deployed it works through the mission plan autonomously. The mission controller and all bound SCS controllers remain passive, but can intervene in case the mission plan is changed, security parameters are violated, or any SCS is taken over by an operator for remote control. After a mission has concluded, the mission controller releases the blocked SCS controllers and is destroyed.

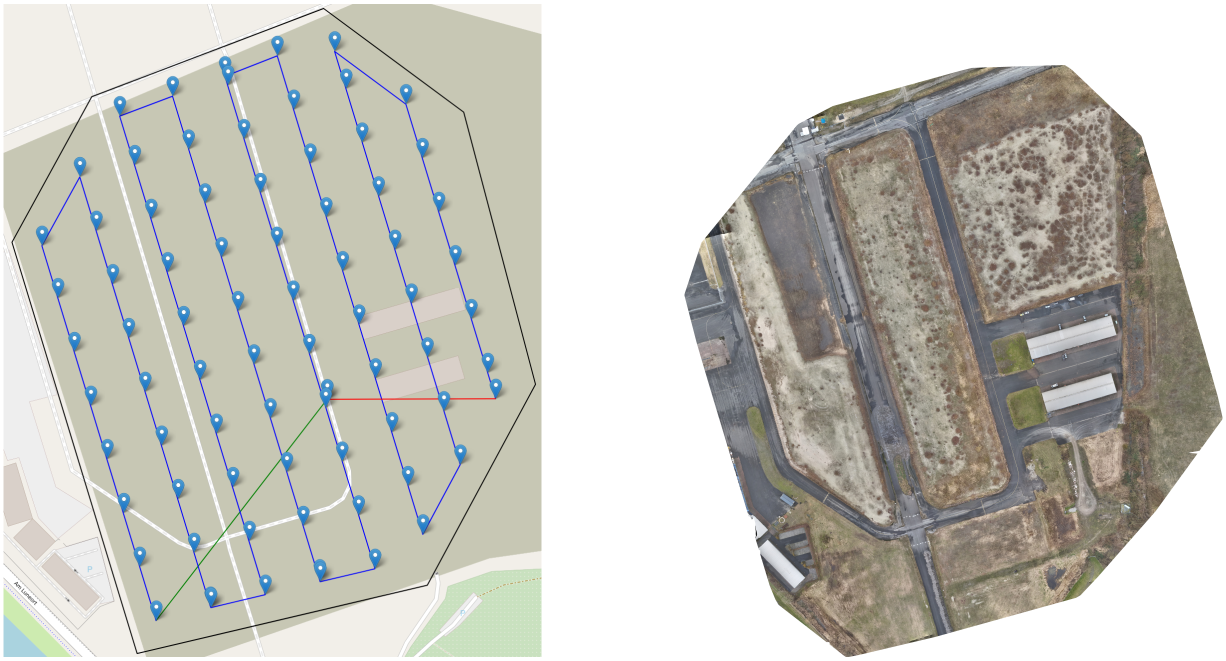
During any mission, the sensor data being sent back to the core is processed in the data processing pipeline and used for updating the map. The data also keeps relevant core nodes and the NEXUS UI up to date.

Within the LSTS toolchain, the communication protocol IMC defines messages for communication between their C2 interface and their robots. These messages cover a large section of message types exchanged during missions to keep all framework components up to date. As such, the IMC protocol will be utilized within NEXUS.

C. Sensor Carrier Pool

The Sensor Carrier Pool (SCP) is comprised of SCSs which are individual mobile robotic platforms providing at

DOI: 10.5281/zenodo.17120080



(a) The mission plan

(b) The resulting orthophoto

Fig. 4: The generated test mission, taking one photo at each waypoint (blue dots in (a)). Approach in green, return in red.

least one sensor data type. The SCSs are able to receive instructions from the core and perform actions accordingly. When deployed, they can alter their behavior depending on their current environment.

We propose using three types of SCS to fully capture the maritime environment: UAVs, USVs, and UUVs. They require hardware and software supporting IMC, mission execution, and behavior adaptation and communication capabilities for missions as swarms. Each SCS should therefore be governed by an LSTS DUNE instance where applicable. As an option, ROS2 can be utilized instead. Eventually, DUNE will need to be extended to enable the wanted levels of SCS autonomy, such as advanced obstacle avoidance behaviors, or specifying an aggressiveness level to allow SCS to devied how individual maneuvers should be executed or replanned.

IV. Current State

For basic architecture validation and interface testing, a skeletal implementation of the NEXUS UI and key NEXUS core nodes was realized. The UI has the capabilities of editing missions and general and task-specific SCS configurations, database interaction with the SCS, mission, and task databases, and allows adding of web-map-service and slippy map tiles sources as basemaps or data overlays. The core's nodes were used to test inter-node communication with custom messages within the framework's structure.

Additionally, a mission generation pipeline example was implemented as a monolithic application for a UAV full coverage survey mission outside the framwork's architecture. Its simplistic nature enables basic planning functionality for waypoint missions, full coverage missions, and missions combining tasks of both types. It generates a uniform occupancy grid map indicating free space from a 2D map excerpt from e.g. Open Street Map [29]. The coverage path is planned utilizing the Rotating Calipers Path Planner [30]. Jump Point Search [31] is used for the approach and return paths and optimized with a simplified any-angle optimization based on [32]. This pipeline was tested by flying a generated mission (Fig. 4a) with a DJI Mavic 3T [33] and creating an orthophoto (Fig. 4b) from the taken photos utilizing OpenDroneMap [34]. The multicoptor was able to follow the specified path by flying from waypoint to waypoint in sequence, controlling the camera payload as specified by the generated mission plan. All waypoints were successfully visited.

V. Conclusion and Future Work

The concept details an approach to provide researchers with a tool for applied research across multiple disciplines without expert knowledge of the individual SCSs. It also will be used directly for in-situ observation of critical maritime infrastructure, thus enhancing situational awareness above and below the water line. Testing the frameworks's definitions with skeleton implementations of key nodes, a first implementation of the NEXUS UI, and a mission generation application showed promise for combining all aspects as detailed by the framework concept. However, to be on par with the basic operating capabilities of frameworks like the LSTS framework the NEXUS Core nodes need to be fully implemented. This includes the transition of the monolithic mission generation pipeline to the node based mission generation pipeline in NEXUS Core. Furthermore, explicit solutions for problems

DOI: 10.5281/zenodo.17120080

like underwater communications or dynamic meshing of individual SCS to create a self-healing communications network need to be addressed. This includes the exact implementation of control software on the SCS to enable a degree of autonomy allowing self-management and mission re-planning on-board, expanding the existing DUNE software or implementing a new solution with e.g. ROS2.

In the following steps, the NEXUS Core nodes supporting the mission generation pipeline will be implemented. The mission and SCS controller nodes, as well as the system manager node will follow. Periodic field tests will accompany the development process, ensuring the framework works as intended. The system interface node must be able to handle the data traffic, and comply to the IMC message protocol. Necessary data processing for the core's basic functionality will enable the core. Lastly, the SCS hardware and software will be integrated for concept verification, practical application, and deployment in real environments.

Once complete, the resulting system will enable crucial research for the safety of critical maritime infrastructures, heterogeneous SCS networks, and provide valuable information for situational awareness.

References

[1] K. Buchholz, "Baltic sea cable incidents pile up," website, 2025. [Online]. Available: https://www.statista.com/chart/33892/damage-to-underwater-cables-and-pipelines-in-the-baltic-sea/

[2] D. Kirichenko, "Ukraine's marauding sea drones bewilder russia," website, 2025. [Online]. Available: https://cepa.org/article/ukraines-marauding-sea-drones-bewilder-russia/

[3] D. Zoldi, "Unmanned aerial vehicles shaping the future of defense in israel, canada and ukraine," website, 2025. [Online]. Available: https://www.autonomyglobal.co/unmanned-aerial-vehicles-shaping-the-future-of-defense-in-israel-canada-and-ukraine/

[4] A. Munir, A. Aved, and E. Blasch, "Situational awareness: Techniques, challenges, and prospects," *AI*, vol. 3, no. 1, pp. 55–77, 2022. [Online]. Available: https://www.mdpi.com/2673-2688/3/1/5

[5] W. DeBusk, "Unmanned aerial vehicle systems for disaster relief: Tornado alley," in *AIAA Infotech@ Aerospace 2010*, 2010, p. 3506.

[6] Laboratório de Sistemas e Tecnologia Subaquática, "Lsts toolchain," website, 2025. [Online]. Available: https://lsts.fe.up.pt/toolchain

[7] J. Pinto, P. S. Dias, R. Martins, J. Fortuna, E. Marques, and J. Sousa, "The lsts toolchain for networked vehicle systems," in *2013 MTS/IEEE OCEANS - Bergen*, 2013, pp. 1–9.

[8] Laboratório de Sistemas e Tecnologia Subaquática, "Ripples," website, 2025. [Online]. Available: https://lsts.fe.up.pt/software/55

[9] J. Pinto, P. S. Dias, and J. B. Sousa, "Ripples: A tool for supervision and control of remote assets," *Instrumentation viewpoint*, vol. 20, 2018.

[10] Laboratório de Sistemas e Tecnologia Subaquática, "Neptus," website, 2025. [Online]. Available: https://lsts.fe.up.pt/software/54

[11] P. Dias, S. Fraga, R. Gomes, G. Goncalves, F. Pereira, J. Pinto, and J. Sousa, "Neptus - a framework to support multiple vehicle operation," in *Europe Oceans 2005*, vol. 2, 2005, pp. 963–968 Vol. 2.

[12] Laboratório de Sistemas e Tecnologia Subaquática, "IMC," website, 2025. [Online]. Available: https://lsts.fe.up.pt/software/63

[13] R. Martins, P. S. Dias, E. R. B. Marques, J. Pinto, J. B. Sousa, and F. L. Pereira, "Imc: A communication protocol for networked vehicles and sensors," in *OCEANS 2009-EUROPE*, 2009, pp. 1–6.

[14] Laboratório de Sistemas e Tecnologia Subaquática, "DUNE," website, 2025. [Online]. Available: https://lsts.fe.up.pt/software/64

[15] MIT, "Welcome to the moos-ivp home page," website, 2025. [Online]. Available: https://oceanai.mit.edu/moos-ivp/pmwiki/pmwiki.php

[16] MIT Laboratory for Autonomous Sensing Systems, "The laboritory for autonomous marine sensing systems (lamss)," website, 2025. [Online]. Available: https://oceanai.mit.edu/lamss/pmwiki/pmwiki.php?n=Main.HomePage

[17] Oxford Robotics Institute, "Oxford robotics institute," website, 2025. [Online]. Available: https://ori.ox.ac.uk/

[18] ArduPilot, "Ardupilot," website, 2024. [Online]. Available: https://ardupilot.org/

[19] PX4, "Open source autopilot," website, 2025. [Online]. Available: https://px4.io/

[20] ArduPilot, "Mission planner home," website, 2025. [Online]. Available: https://ardupilot.org/planner/

[21] Q-Ground Control, "Qgroundcontrol," website, 2025. [Online]. Available: https://qgroundcontrol.com/

[22] MAVLink, "Mavlink developer guide," website, 2025. [Online]. Available: https://mavlink.io/en/

[23] ROS, "Ros- robot operating system," website, 2025. [Online]. Available: https://www.ros.org/

[24] MRPT, "Mrpt," website, 2025. [Online]. Available: https://docs.mrpt.org/reference/latest/index.html

[25] YARP, "Yarp: Yet another robot platform," website, 2025. [Online]. Available: https://www.yarp.it/latest/

[26] Angular, "Angular," website, 2025. [Online]. Available: https://v19.angular.dev/overview

[27] Eprosima, "Fast dds," website, 2025. [Online]. Available: https://fast-dds.docs.eprosima.com/en/stable/

[28] ZeroMQ, "Zeromq - an open-source univeral messaging library," website, 2025. [Online]. Available: https://zeromq.org/

[29] O. Contributors, "Open street map," website, 2025. [Online]. Available: https://www.openstreetmap.org/about

[30] J. I. Vasquez-Gomez, M. Marciano-Melchor, L. Valentin, and J. C. Herrera-Lozada, "Coverage path planning for 2d convex regions," *J. Intell. Robotics Syst.*, vol. 97, no. 1, p. 81–94, Jan. 2020. [Online]. Available: https://doi.org/10.1007/s10846-019-01024-y

[31] D. Harabor and A. Grastien, "Online graph pruning for pathfinding on grid maps," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 25, no. 1, pp. 1114–1119, Aug. 2011. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/7994

[32] Y. Luo, J. Lu, Q. Qin, and Y. Liu, "Improved jps path optimization for mobile robots based on angle-propagation theta* algorithm," *Algorithms*, vol. 15, no. 6, 2022. [Online]. Available: https://www.mdpi.com/1999-4893/15/6/198

[33] DJI, "Dji mavic 3 enterprise serie," website, 2025. [Online]. Available: https://enterprise.dji.com/de/mavic-3-enterprise

[34] OpenDroneMap Authors, "Odm – a command line toolkit to generate maps, point clouds, 3d models and dems from drone, balloon or kite images." website, 2020. [Online]. Available: https://www.opendronemap.org/

DOI: 10.5281/zenodo.17120080